NAME

scm -- scm language

DESCRIPTION

The *scm* textual language was introduced in the Ph.D. thesis of Tristan Le Gall. An *scm* model contains the description of a system of communicating machines. This description is composed of two parts:

- A *header* that specifies the set of fifo channels and the message alphabet.

- A list of *automata*, each modeling a communicating machine.

For model-checking purposes, the format also permits the specification of a set of *bad configurations*.

Header

1. Start the description of a system of communicating machines and specify its name:

   scm *ident* **:**

2. Specify the number of channels:

   nb_channels = *integer* **;**

   The set of channels is {0, ..., *n*-1} where *n* is the number of channels. By default, all channels are perfect.

3. Tag some channels as lossy (optional):

   [**lossy :** *integer* [**,** *integer* ...]]

4. Declare the message alphabet as follows:

   parameters : [{**int** | **real**} *ident* [= *expr*] **;** ...]

   The alphabet is the same for all channels. Each message holds a typed numerical value.

Automata

1. Start the description of an automaton and specify its name:

   automaton *ident* **:**

2. Declare the automaton’s local variables (optional):

[{**int** | **real**} *ident* [**=** *expr*] **;** ...]

3. Specify the automaton's initial states:

   initial : *integer* [**,** *integer* ...]

4. Declare the automaton's states together with outgoing transitions:

   state *integer* **:** [**to** *integer* **:** *command* **;** ...]

   where *command* is of the following form:

   when *cond* [**,** *integer* {**!** | **?**} *ident*] [**with** *ident* **=** *expr* [**,** *ident* **=** *expr* ...]]

Semantics

The operational semantics of a system of communicating machines is the usual one: the automata move asynchronously according to their local transitions, and they communicate exclusively through the channels. Communication actions are **!** (send) and **?** (receive). Channels are fifo, unbounded, and initially empty. Note that channels need not be point-to-point, they are shared by all automata.

Bad Configurations

The description of the system of communicating machines is, optionally, followed by the specification of a set of bad configurations:

bad_states : (automaton *ident* **:** *badlocal* ... [**with** *regexp*] **)** ...

where *badlocal* is of the following form:

in *integer* **:** *cond* ...

The local constraints in a *badlocal* specification are *disjuncted* together. The *badlocal* specifications in a **bad_states** declaration are *conjuncted* together.

The symbol **#** is used to separate channels in *regexp*. Each word matched by *regexp* must contain exactly *n*-1 occurrences of **#** where *n* is the number of channels.

SEE ALSO

control(1), verify(1)